Optimizing Spacecraft Design – Optimization Engine Development: Progress and Plans

Steven L. Cornford, Martin S. Feather, Julia R. Dunphy, Jose Salcedo
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr
Pasadena, CA 91109
818-354-1701, 818-354-1194, 818-393-5365, 818-393-0995
{Steven.L.Cornford,, Martin.S.Feather, Julia.R.Dunphy, Jose.Salcedo}@Jpl.Nasa.Gov

Tim Menzies
Lane Department of Computer Science & Electrical Engineering,
West Virginia University
PO Box 6109, Morgantown, WV 26506-6109
304-367-8263
tim@menzies.com

Abstract— At JPL and NASA, a process has been developed to perform life cycle risk management. This process requires users to identify: goals and objectives to be achieved (and their relative priorities), the various risks to achieving those goals and objectives, and options for risk mitigation (prevention, detection ahead of time, and alleviation). Risks are broadly defined to include the risk of failing to design a system with adequate performance, compatibility and robustness in addition to more traditional implementation and operational risks. The options for mitigating these different kinds of risks can include architectural and design choices, technology plans and technology back-up options, test-bed and simulation options, engineering models and hardware/software development techniques and other more traditional risk reduction techniques.

Each of these risk mitigations has resource costs associated with them. The sum of all these mitigations is almost always unaffordable. Furthermore, there may be a variety of other constraints (mass, power, funding profile, leveraged programs, etc.) that further constrain acceptable selections. The challenge is therefore to emerge with an "optimal" selection of mitigations that makes best use of available resources to reduce risk to the fullest extent possible.

For non-trivial design spaces, the search space of possible selections is huge. This precludes exhaustive search for the optimum, and therefore necessitates the adoption of heuristic search techniques. At JPL, we have explored application of several heuristic techniques for searching for, and refining, collections of risk mitigations, notably: genetic algorithms, simulated annealing, and machine learning. The results of research and pilot applications of these techniques for finding best combinations of life cycle risk management solutions are discussed.

TABLE OF CONTENTS

1. DDP'S RISK MANAGEMENT PROCESS[-]

This section summarizes the risk management process that we have developed and applied at JPL and NASA.

Defect Detection and Prevention (DDP) is the risk management process that we have developed and applied to risk assessment, risk mitigation planning, and lifecycle risk management [1]. The primary purpose of DDP is to help expert users plan the design and development of complex systems. Risk management is central to their successful development, deployment and operation. Custom tool support [2] facilitates the practical application of the DDP process.

DDP explicitly represents risks, the objectives that risks threaten, and the mitigations available for risk reduction. By linking these three concepts, DDP is able to represent and reason about the cost-effectiveness of risk reduction

- Paper 1119
- 0-7803-7651-X/03/$17.00 © 2003 IEEE

alternatives. In more detail, the DDP representation includes:

Objectives – the requirements, goals and objectives that the system is to achieve. Each of them has a "weight" representing its relative importance.

Risks – all the possible things that, should they occur, would detract from attainment of the Objectives. It is important to realize that in applying DDP, this category of information is quite broad. For example, DDP uses Risks to encompass design risks (failing to design a system with adequate performance, compatibility and/or robustness) in addition to more traditional implementation and operational risks.

Mitigations – all the possible activities that, if they are performed, will reduce risk. Again, DDP uses this category of information very broadly. For example, DDP uses Mitigations to encompass architectural and design choices, technology plans and technology back-up options, test-bed and simulation options, engineering models and hardware/software development techniques in addition to more traditional risk reduction techniques. Mitigations have costs – the resources it takes to apply them. These might include monetary, schedule, physical resources (e.g., mass, power), facilities (e.g., test equipment), etc.

Impacts – each of these quantifies the extent to which a risk, should it occur, detracts from the attainment of an objective. **Effects** – each of these quantifies the extent to which a mitigation, should it be applied, reduces a risk.

2. OPTIMIZATION NEEDS

In the JPL and NASA setting, spacecraft systems are the focus. We have applied DDP to help plan the development of individual technologies (both hardware and software) for use on spacecraft, and, in ongoing work, are using DDP to help in the planning for an entire spacecraft.

The primary purpose of DDP is to assist users to cost-effectively select risk mitigations. Each mitigation has resource costs. The sum total cost of all these mitigations is almost always unaffordable. Furthermore, there may be a variety of other constraints (mass, power, funding profile, leveraged programs, etc.) that further constrain acceptable selections. The challenge is therefore to emerge with an "optimal" selection of mitigations that makes best use of available resources to reduce risk to the fullest extent possible.

In these applications, objectives, risks and mitigations are numerous, and highly interlinked, making optimization a challenge. To convey the magnitude of this challenge, the topology of the DDP information for a recently concluded study of a packaging technology is shown in Figure 1. The study gathered 50 Objectives, 31 Risks and 58 Mitigations. In the figure, the Objectives are shown as small squares in the row across the top, the Risks as small squares in the row across the middle, and the Mitigations as small squares in the row across the bottom are the Mitigations. The lines connecting these indicate the presence of the Impacts (between Risks and Objectives) and Effects (between Mitigations and Risks). This is just the topology – there is further quantitative information not apparent on this figure, most importantly the strengths of these Impacts and Effects, nor the relative weights of the Objectives.

For non-trivial DDP applications, the search space of possible selections is huge. For example, in the application whose data is pictured in Figure 1, there are 58 individual mitigations, so the number of possible selections of such is 2^{58} (approximately 10^{18}). In other DDP applications, numbers of mitigations have been comparable, or even greater, with correspondingly larger search spaces. This makes cost-effectively selecting mitigations a considerable challenge. What is needed is the ability to automatically optimize selection of Mitigations, as called for in [3].

In response, we have been investigating the use of heuristic search techniques. These have the capacity to locate near-optimal solutions in such huge search spaces. In the DDP setting we have experimented with three different kinds of heuristic search techniques: genetic algorithms, machine learning, and simulated annealing. These are discussed in detail in the sections that follow.

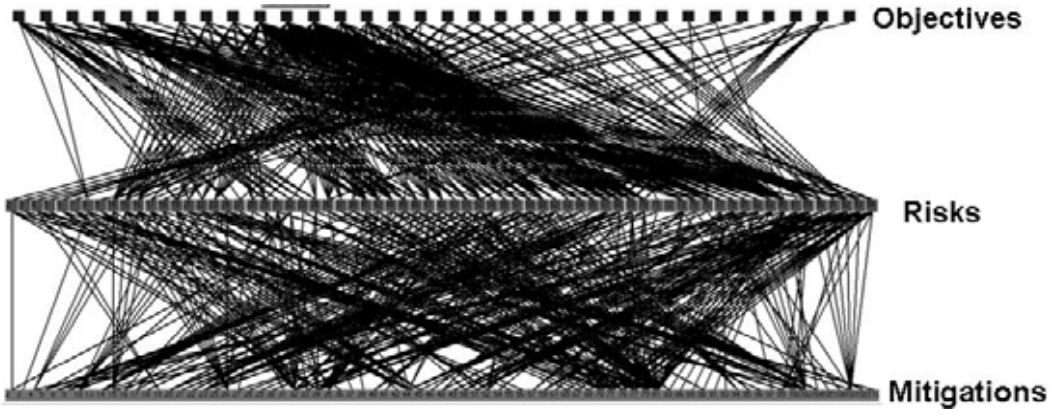

Figure 1 – The Linked Structure of Information in a Typical DDP Application

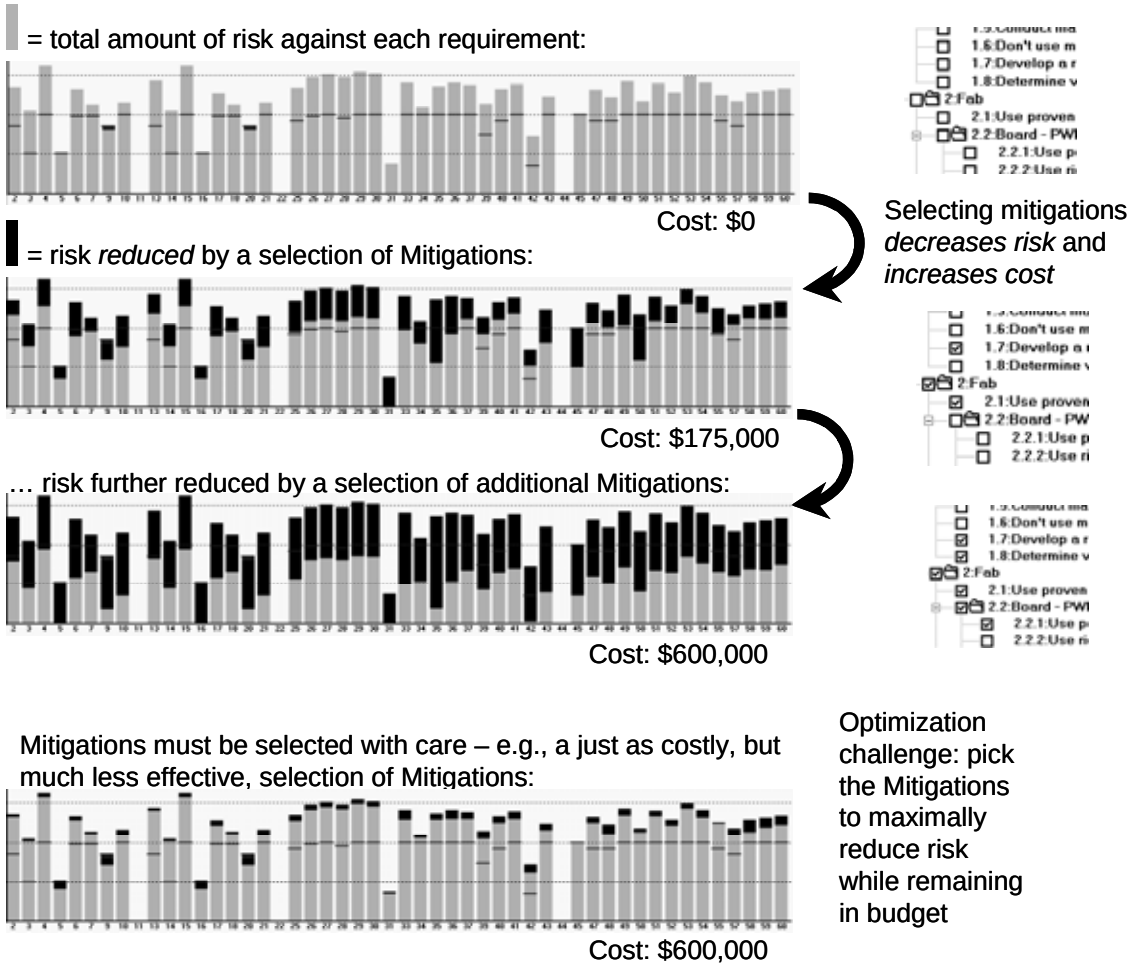

Figure 2 – The optimization challenge: judiciously select mitigations to cost-effectively reduce risk

3. GENETIC ALGORITHM EXPERIMENTS

Our initial experiments were with Genetic Algorithms used to search for DDP risk-minimizing solutions for a given cost ceiling.

The core idea of Genetic Algorithms is to work with a population of candidate solutions. This population is allowed to evolve in a series of steps. In each step, the better solutions, and variations of them, are preferentially favored in populating the next generation. Various forms of "mutations" are used to generate those variations.

In a DDP application, a candidate solution is represented as a vector of Booleans, one for each of the Mitigations of that application. The Boolean is true if the Mitigation is to be selected for application in the solution, false otherwise. A population of solutions is thus a set of such vectors. DDP calculates the risk for a given solution by summing the impact of the Risks on the weighted Objectives, taking into account the risk-reducing effects of the Mitigations selected by that solution. This risk measure is used as the score for the Genetic Algorithm (where lower risk is better).

To apply Genetic Algorithms, we devised a way to generate mutations from a given solution such that those solutions would each fall within the prescribed cost ceiling. In generating the next generation of candidates from the current generation, the better-scoring solutions of the current generation were allowed to contribute more such (cost-capped) mutations to the next generation. Over a series of generations, the search would tend to discover better solutions, i.e., selections of Mitigations whose total costs remained below the cost ceiling, and whose risk-reducing effects were superior.

The experiments with this approach demonstrated the following strengths and weaknesses:

Genetic Algorithm: Application Strengths

Rapid Progress – the algorithm made rapid progress at finding much improved solutions. This held true for quite large DDP datasets. We observed this in experimenting extensively with data from a relatively large DDP application, one that had 99 Mitigations from which to select.

We attribute this performance to the work we put into generating mutations (of solutions in the current generation) that remained within the cost ceiling. This considerably reduced the search space to just those Mitigation selections

that exhibited the required cost characteristic.

This effect became even more pronounced when we took into account additional constraints on costing. These constraints arose in the context of planning risk mitigation for a long-duration project. The project not only had a constraint on the sum total cost for the project as a whole, but also a constraint on the sum total cost within each time phase (financial year). The additional constraints that this placed on solutions further reduced the search space. We extended our mutation algorithm accordingly, so that it would generate those and only solutions that met all of these costs constraints. As a result, the performance of the Genetic Algorithm search was improved.

Multiple Solutions – a desirable feature of the Genetic Algorithm approach is that in each step it calculates *multiple* candidate solutions. This means that users can examine the better scoring solutions from the final generation, and pick from among them. When there are several solutions that have close cost and risk scores, minimizing the risk is not necessarily the preferred way to narrow the choice to just one solution. Instead, users often wish to take other factors into account to guide their selection.

Genetic Algorithm: Application Weaknesses

Optimize for Benefit (Least Risk) Only – the way we approached the use of Genetic Algorithms would not work to optimize for cost minimization at a given level of risk. This is because the calculation of risk in a DDP model is much more convoluted than the calculation of cost. As a consequence, that generation of mutations to each fall below a risk ceiling is far harder than the generation of mutations to each fall below a cost ceiling.

An alternative would be to allow mutation to generate solutions unconstrained by a benefit ceiling. During the population of the next generation from the current one the risk measure would be taken into account – those current solutions that exceed the benefit ceiling would be scored as worse than those solutions that are within the benefit ceiling, regardless of cost. Unfortunately, this would detract from the rapid progress quality that our cost-capped approach exhibited.

Hard to Maintain as DDP Cost Model is Elaborated – the way we approached cost-capped generation of mutations worked well with the mainstream DDP cost model. However, we have subsequently elaborated that cost model to better represent cost phenomena that arise in practice. Elaborating our mutation generation algorithm accordingly would be troublesome.

The key such elaboration of the DDP Cost Model is to handle repair costs – these are incurred by DDP mitigations that *detect* (rather than prevent or alleviate) risks. Their benefit comes from exposing the presence of risks, thus allowing for corrective measures to be taken prior to actual use of the system. The net result is that the operational system (in our case, the spacecraft post-launch) is more reliable. However, two kinds of costs are incurred: the cost of performing the detection-style mitigation, and the cost of repairing the problems (if any) that it detects. The calculation of the expected repair cost depends on when it is done (repair costs typically escalate later in the life-cycle) and how likely it is that the risk will be present at the time the detection is performed. This likelihood in turn depends on what other mitigations have preceded it. Unfortunately, this intertwining of the cost calculation with the risk calculation renders generation of cost-capped mitigations a challenge, in much the way that generating risk-capped mitigations would be.

Prior to this elaboration of the DDP model we approximated the situation by ascribing appropriately high costs to the mitigations that we knew to be late-lifecycle detections. This approximation suffices in many of the decision-making challenges we tackle with DDP. However, the elaborated cost model's more accurate representation of cost factors is desirable to have. For example, we have used it in example calculations of cost-benefit measures for alternative software quality practices – see [4] for details.

Tuning Genetic Algorithms' Search – Genetic Algorithms often require tuning to get efficient search performance. While they offer a plethora of parameters for such tuning (e.g., to control relative rates of various kinds of mutations), we do not want to have to burden the DDP user with these choices. To date we have not performed experiments on a broad enough range of DDP applications to say how much of a problem this will be for us. We would have to address this issue were we to incorporate a Genetic Algorithm optimizer as part of DDP.

Can't Distinguish Critical from Non-Critical Decisions – the recommended selection of mitigations that is the end product of a Genetic Algorithm run on a DDP application contains no indication of the relative importance of individual decisions. Lacking this information, users do not know which of those decisions they can safely modify without severely compromising the quality of the solution. Of course, they can use the DDP tool to explore, making adjustments and studying the recalculated cost and benefit measures, but this is a tedious process at best.

4. MACHINE LEARNING EXPERIMENTS

Classical machine learning (e.g. C4.5 [5]) can be applied to learn from examples the implications between sets of decisions and results. For a model such as DDP, individual decisions are binary (which Mitigations to apply), and results are measures of cost and benefit (reduction of Risk, or equivalently, attainment of Requirements). However, as pairs, triples, etc., of individual decisions are considered, the numbers increase, and some of summarization is required. Our co-author Tim Menzies has pioneered the use

of an approach that combines learning and summarization into one tool, his "TAR2" system. We used in experiments on DDP applications. The approach and results are explained at length in [6]. Briefly, the way we combined DDP and TAR2 is sketched in Figure 3.

DDP is used to generate a large number of examples, where in each example the Mitigations to apply are chosen at random. A given example's information comprises the choices of selected Mitigations, and a score reflecting the net value of the cost and benefit (requirements attainment) of those selections. The generated set of such examples is passed over to TAR2. From the examples, TAR2 determines which are the most critical decisions, and how to make them so as to maximize the score. TAR2 actually determines several alternative such decisions sets, not just one. This is an opportune point for the human experts to be involved – they can look at the alternative decision sets, and select the one they would most prefer. Having made such a selection, the information is passed back to DDP, and the cycle repeats. In this second and subsequence cycles, the decisions made in the prior cycles constraint DDP's choices in its generation of the next set of examples. This process is continued until convergence (either all the possible decisions have been made, or there is so little variation from the decisions that remain that there is no need to continue further).

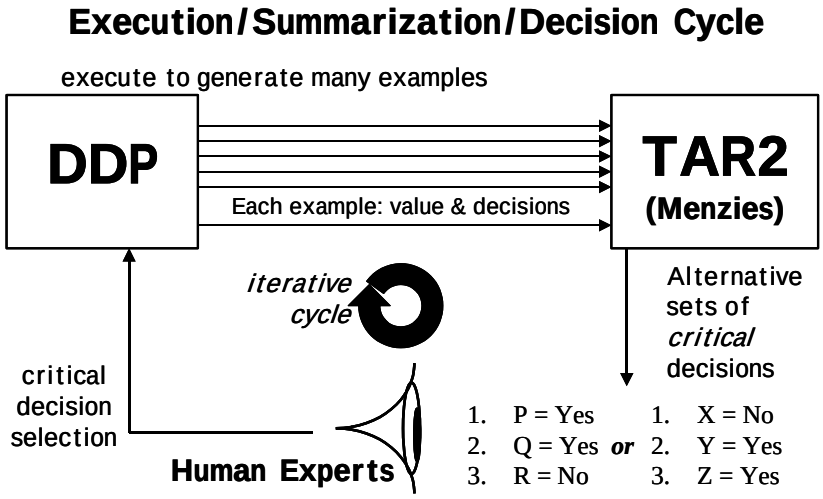

Figure 3 – The Combination of DDP and TAR2

This approach was piloted on data from a large-scale DDP application. The application had 99 Mitigations from which to select. Five cycles of the DDP-TAR2 iteration described above were sufficient to identify the 33 most critical decisions (some of these were decisions of which Mitigations to perform, the others of which Mitigations to *not* perform). Making just these 33 decisions (leaving the other 66 to be made at random) was sufficient to constrain the resulting solutions to a very compact region of the search space in the optimal corner. This is shown graphically in Figure 4. Each black point on the chart is one of the randomly generated examples (selection of Mitigations). Its placement is determined by its DDP-computed cost (placement on horizontal axis), and benefit, i.e., attainment of objectives (placement on vertical axis). The dispersal of these black points indicates that when Mitigations are chosen at random, there is a large variation in the quality of their solutions. The compact white area is comprised of a large number of closely spaced white points, each corresponding to one of the solutions whose 33 most critical decisions were constrained as directed by TAR2. The remaining variation within this area is due to the random choice from the remaining 66 less critical decisions.

From this figure it is clear that the iterative use of TAR2 has located near-optimal solutions (those towards the upper left corner). In each of the TAR2-DDP iterations the additional decisions made by TAR2 nudge the search focus towards the optimal area, allowing it to explore that area much more effectively than would have been possible with random search alone.

TAR2 Machine Learning: Application Strengths

Identifies the Most Critical Decisions – the use of TAR2 identifies which of the many decision are the most critical. In DDP terms, each decision is whether or not to apply a Mitigation. Those to be applied are the Mitigations that contribute the most towards an overall cost-effective solution, while those to *not* be applied are the Mitigations that detract the most from an overall cost-effective solution. The remaining Mitigations make little net difference. Note that the interconnectedness typical of DDP applications (recall Figure 1) makes identification of these decisions far from easy. For example, a Mitigation may appear to be cost-effective when considered in isolation (because it reduces many risks and costs relatively little), yet, in the context of the other Mitigations, be redundant (because Mitigations that must be applied to reduce other risks happen to also reduce the risks that this promising Mitigation addresses).

Multiple Solutions – at each application of TAR2 in the iterative DDP-TAR2 cycle, TAR2 proposes several alternative sets of decisions. This gives users the opportunity to select from among them, allowing them to inject their preferences into the search process. Coupled with the fact that TAR2 identifies the most critical decisions first, this means that users' attention and decision making is being applied where it matters the most. This is a very effective blend of automated search and expert human guidance.

Robust Interface between TAR2 Search and DDP Risk Model – the interface between TAR2 and DDP is very straightforward, meaning that as DDP evolves, little or no change needs to be made to TAR2. All that TAR2 needs is a set of examples, generated and scored by DDP. In return, TAR2 provides back to DDP a set of decisions. The elaboration of the DDP cost model we discussed earlier requires no change to TAR2 – it continues to operate as before, relying upon DDP to correctly score each of the examples it generates based on the internal-to-DDP model of cost and benefit.

Essentially, TAR2 is uncoupled from the innards of the DDP cost/benefit model. This decoupling has permitted TAR2 to be applied in a number of widely differing domains, including the COCOMO risk model [7] and CMM level 2 [8].

The discussion of why TAR2 succeeds in such widely differing applications is beyond the scope of this paper – the interested reader is referred to [9] and [10] for further details.

Optimize for Cost and/or Benefit (Least Risk) – the TAR2 form of optimization can be applied to search for benefit maximization (maximal attainment of Objectives, or equivalently, minimal Risk) within a given cost ceiling, cost minimization within a given Risk ceiling, or a weighted combination of the two. The key to this is the use of the DDP-computed score for each example passed over from DDP to TAR2. The calculation of that score embodies the optimization goal, allowing TAR2 to search for solutions that achieve that goal, but decoupling it from the details of the DDP model calculations.

TAR2 Machine Learning: Application Weaknesses

Slow Progress – in each iteration TAR2 requires of DDP the generation and scoring of the large number of examples; this is a slow process. The bottleneck lies in the DDP side, where the risk model itself is relatively complex. Requiring that tens of thousands of examples each be scored is time consuming. The multiple cycles required by the DDP-TAR2 connection exacerbates this problem.

Manual control of TAR2 required – in the version of TAR2 used in the pilot study, some manual control of TAR2 was

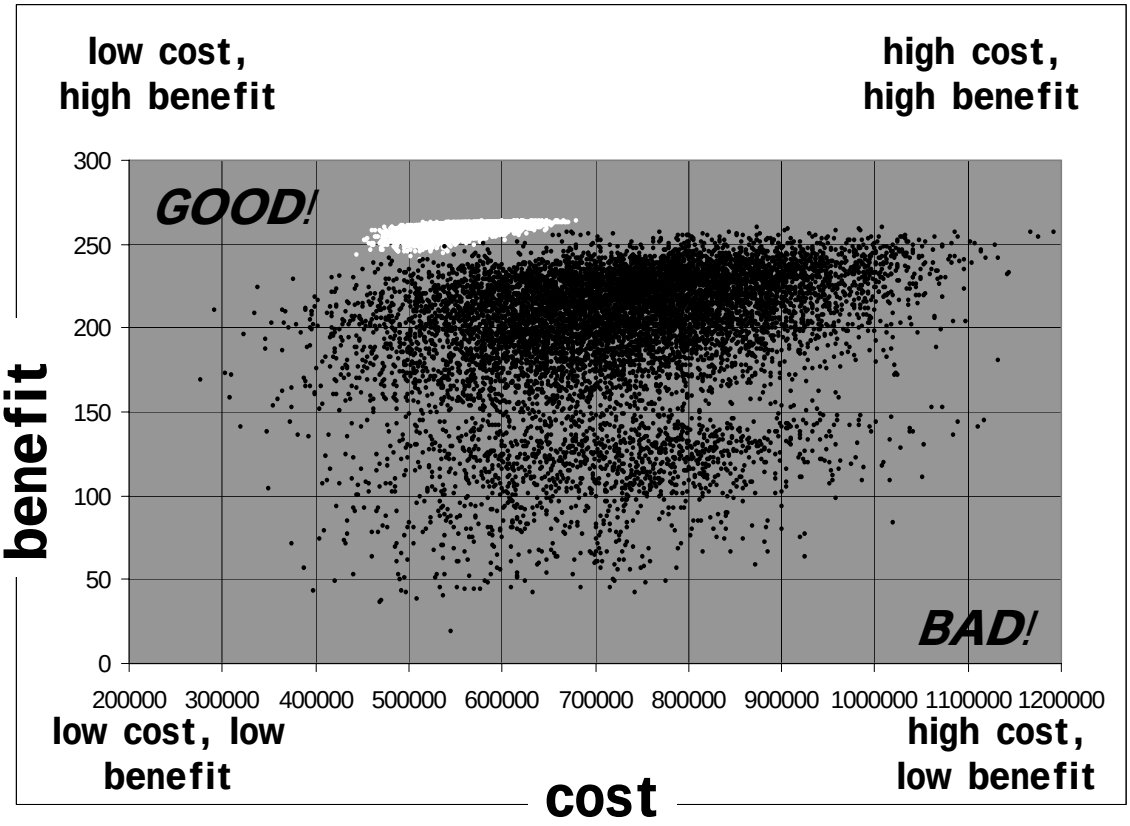

Figure 4 – TAR2's Search Results on a DDP Application

required in each cycle. This was relatively easy to learn how to do – Tim Menzies taught one of the DDP team, so it did not require the intimate involvement of the TAR2 creator to apply the TAR2 tool. Nevertheless, this would be an impediment to the incorporation of TAR2 as the optimizer part of DDP.

5. Simulated Annealing Experiments

Our final category of heuristic search experiments used Simulated Annealing.

The core idea of Simulated Annealing is a search that favors improving the best solutions found so far. However, in order to better explore the space, it allows the search to take retrograde steps (to worse solutions). Tolerance for such retrograde steps is gradually decreased as the search progresses. The net result is rapid convergence towards near-optimal solutions.

Simulated Annealing requires the ability to score a solution, and the ability to step from the current solution to a candidate next solution. Recall that in a DDP application, a candidate solution is represented as a vector of Booleans, one for each of the Mitigations of that application. The Boolean is true if the Mitigation is to be selected for application in the solution, false otherwise. Thus to apply Simulated Annealing to DDP, the score of a solution is derived from the DDP-calculated cost and benefit. The goals of the optimization determine how this score is calculated. For example, if the goal is to select Mitigations that lead to maximal attainment of Objectives while remaining within a given cost ceiling, the score for a solution would be higher the greater the attainment of Objectives, as long as the cost was below the cost ceiling. A solution whose cost is above the cost ceiling would be scored very low. The ability to step from the current solution to a candidate next solution is simply the random modification of a percentage of the current solution's vector of Booleans representing "selectedness" of Mitigations.

Figure 5 shows a run of the Simulated Annealing algorithm on an actual DDP dataset. The cost ceiling has been set at $1,000,000, indicated by the position of the vertical line. Points to the right of the line exceed the cost ceiling. The run is able to locate solutions (Mitigation selections) that achieve close to the maximal possible benefit (attainment of Objectives) while remaining below the cost ceiling. During the course of the search the effect of simulated annealing is to decrease tolerance for retrograde steps (the analogy with the physical phenomenon of simulated annealing is of the temperature "cooling"). Note that the plot is of *all* the candidate solutions examined by the search, many of which exceed the cost ceiling. Their points are plotted, but they generally do not become the current working candidate during search. As the search progresses, we keep track of the best solutions found to date. Once a search has completed, the user may view the selections of Mitigations that the search found.

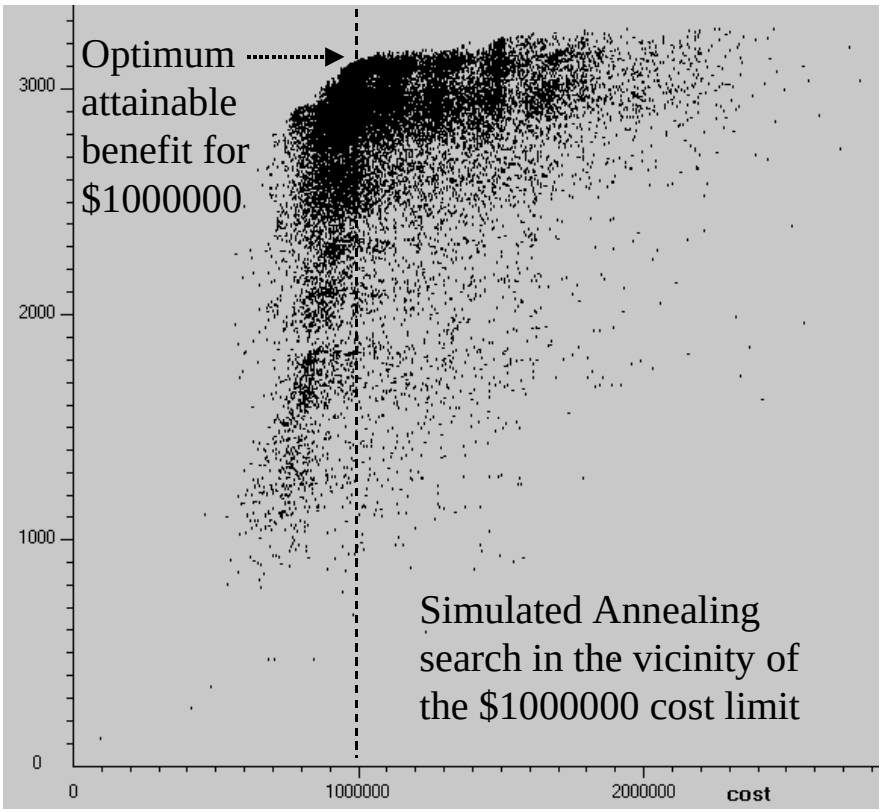

Figure 5 – Simulated Annealing on a DDP Application

This and other experiments with Simulated Annealing demonstrated the following strengths and weaknesses:

Simulated Annealing: Application Strengths

Rapid Progress – the algorithm made rapid progress at finding much improved solutions. We attribute this to the fundamental strength of Simulate Annealing style search. Because the search deals with only one solution at a time, it is expedient to explore a large number of steps. The example in Figure 4 was a 10,000-step search, which took approximately 20 minutes running on a 1.8 MHz laptop. The bulk of the time is spent in DDP's evaluation the cost-benefit score for each candidate solution.

Robust Interface between Simulated Annealing Search and DDP Risk Model

– the interface between Simulated Annealing and DDP is very straightforward. As was the case with the Machine Learning experiments, the reason for this is that the search is uncoupled from the innards of the DDP cost/benefit model.

Optimize for Cost and/or Benefit (Least Risk) – the Simulated Annealing form of optimization can be applied to search for benefit maximization (maximal attainment of Objectives, or equivalently, minimal Risk) within a given cost ceiling, cost minimization within a given Risk ceiling, or a weighted combination of the two. As was the case with the Machine Learning experiments, the reason for this is that the search is uncoupled from the innards of the DDP cost/benefit model.

Automatic Search – the Simulated Annealing search requires no user intervention once it is underway. Prior to starting the search the user must indicate the optimization goals (e.g., the cost ceiling for a cost-constrained optimization), and select the number of steps of search, but there is no need to further tune the search process, or intervene as it proceeds.

Simulated Annealing: Application Weaknesses

Can't Distinguish Critical from Non-Critical Decisions – the recommended selection of mitigations that is the end product of a Simulated Annealing run on a DDP application contains no indication of the relative importance of individual decisions. This is the same weakness as the Genetic Algorithm application approach.

Lack of Multiple Solutions – the Simulated Annealing search works with a single candidate solution. Thus the end result of a search is the best solution found to date, not a set of multiple leading solutions. We have partially addressed this weakness by adjusting our Simulated Annealing algorithm to keep track of the N best solutions to date, the value of N set in advance by the user. However, when there are trivial variants of the best solution, this solution is prone to being swamped by those small variants, and loses track of other solutions that do not score quite so well but are significantly different alternatives to risk reduction.

6. STATUS AND FUTURE WORK

The current status of our work is that the Simulated Annealing based search has become a part of the standard DDP distribution. It has already been used to good effect in a recently conducted DDP study of an advanced packaging technology.

We have augmented our Simulated Annealing search algorithm to take into account user defined constraints in addition to the cost ceiling (or risk ceiling) goals already discussed. An example is that two Mitigations may be mutually exclusive. In principle it would be possible to encode this information directly in the cost scoring function (making any solution that applies both of them score as infinite cost), but this is not an intuitive way to specify such constraints. Instead, the DDP interface proffers a set of typical constraint templates, that users can instantiate as needed to the study at hand. DDP then automatically folds these constraints into the scoring algorithm, so that candidate solutions violating one or more such constraints score poorly. For small numbers of such constraints, this solution appears to work well.

In the future we seek to combine the advantages exhibited by each of the heuristic search techniques that we have explored. We are particularly interested in combining the speed of the Simulated Annealing search with the ability of the TAR2 Machine Learning to identify which are the most critical decisions. We also foresee the need to handle more complex sets of user-defined constraints. Our desire is to be able to automatically fold these into the search generation phase (e.g., for Simulated Annealing, the step to the next candidate solution). This would achieve the effect of a more rapid search because it would not stray into infeasible solution regions. We are also interested in using search to reveal the overall solution space, and "interesting" options therein. For example, the user might have asked to optimize for benefit (maximal Objectives attainment) within a given cost ceiling, but if the search process can recognize that for only a small increment in cost, vastly superior benefits are attainable, it would be desirable to recognize this, and report it to the user.

Our experiments show that heuristic search techniques can work well on real DDP application datasets. The positive reaction to the use of Simulated Annealing based search within a real study suggests that these techniques have a valuable role to play. That we are seeing benefit from our relatively preliminary use of Simulated Annealing based search is very encouraging.

ACKNOWLEDGEMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

The elements of this research have been conducted within the Code Q funded Failure Detection and Prevention Program (FDPP), the Code Q funded Advanced Risk Reduction Task (ARRT) managed at the NASA IV&V facility, and the Code R funded Engineering for Complex Systems Program.

Discussions with JPLers Ken Hicks, Kelly Moran, Steve Prusha and Burton Sigal (JPL) have been most useful in helping us formulate our ideas and bring them to fruition.

REFERENCES

[1] S.L. Cornford, M.S. Feather and K.A. Hicks: "DDP – A tool for life-cycle risk management", *Proceedings, IEEE Aerospace Conference*, Big Sky, Montana, Mar 2001, pp. 441-451.

[2] M.S. Feather, S.L. Cornford and M. Gibbel: "Scalable Mechanisms for Requirements Interaction Management", *Proceedings 4th IEEE International Conference on Requirements Engineering*, Schaumburg, Illinois, 19-23 Jun 2000, IEEE Computer Society, pp 119-129.

[3] S.L. Cornford, J. Dunphy and M.S. Feather: "Optimizing the Design of end-to-end Spacecraft Systems using failure mode as a currency", *IEEE Aerospace Conference*, Big Sky, Montana, 2002.

[4] M.S. Feather, B. Sigal, S.L. Cornford & P. Hutchinson: "Incorporating Cost-Benefit Analyses into Software Assurance Planning", *Proceedings, 26th IEEE/NASA Software Engineering Workshop*, Greenbelt, Maryland November 27-29, 2001.

[5] R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufman, 1992.

[6] M.S. Feather, M.S. & T. Menzies: "Converging on the Optimal Attainment of Goals", Proceedings *of the IEEE Joint International Conference on Requirements Engineering*, Essen, Germany, Sept. 2002, pp. 263-270.

[7] T.J. Menzies and E. Sinsel: "Practical Large Scale What-if Queries: Case Studies with Software Risk Assessment", *Proceedings 15th IEEE International Conference on Automated Software Engineering*, Grenoble, France, Sept 2000, pp 165-173.

[8] T. Menzies and J.D. Kiper: "Better reasoning about software engineering activities", *Proceedings 16th International Conference on Automated Software Engineering*, San Diego, California, Nov 2001, pp 391-394.

[9] T. Menzies, S. Easterbrook, B. Nuseibeh and S. Waugh: "An Empirical Investigation of Multiple Viewpoint Reasoning in Requirements Engineering", *Proceedings 4th IEEE International Symposium on Requirements Engineering*, 1999.

[10] T. Menzies and H. Singh: "Many Maybes Mean (Mostly) the Same Thing", *2nd International Workshop on Soft Computing applied to Software Engineering* (Netherlands), Feb. 2001.

BIOGRAPHIES

Steven Cornford *is a Senior Engineer in the Strategic Systems Technology Program Office at NASA's Jet Propulsion Laboratory. He graduated from UC Berkeley with undergraduate degrees in Mathematics and Physics and received his doctorate in Physics from Texas A&M University in 1992. Since coming to JPL he focused his early efforts at JPL on establishing a quantitative basis for environmental test program selection and implementation. As Payload Reliability Assurance Program Element Manager, this evolved into establishing a quantitative basis for evaluating the effectiveness of overall reliability and test programs as well as performing residual risk assessments of new technologies. This has resulted in the Defect Detection and Prevention (DDP) process is the motivation for this paper. He received the NASA Exceptional Service Medal in 1997 for his efforts to date. He has been an instrument system engineer, a test-bed Cognizant Engineer and is currently involved with improving JPL's technology infusion processes as well as the Principal Investigator for the development and implementation of the DDP software tool. Steve is the one in the middle of the picture.*

Martin Feather *is a Principal in the Software Quality Assurance group at JPL. He works on developing research ideas and maturing them into practice, with particular interests in the areas of software validation (analysis, test automation, V&V techniques) and of early phase requirements engineering and risk management. He obtained his BA and MA degrees in mathematics and computer science from Cambridge University, England, and his PhD degree in artificial intelligence from the University of Edinburgh, Scotland. Prior to joining JPL, Dr. Feather worked on NSF and DARPA funded research while at the University of Southern California's Information Sciences Institute. For further details, see http://eis.jpl.nasa.gov/~mfeather*

Julia Dunphy *received her Master's and Bachelor's degrees in Physics and Mathematics from Cambridge University, UK, ('63) and her doctorate in Theoretical Physics from Stanford in '67. After a career in magnetic recording research in the 70s, she switched to software development and was a cofounder of a small company which provided development software for the then infant microcomputing industry. She now works as a contractor to JPL in the areas of design research and network computing. Her interests include the field of collaborative engineering design infrastructures and automatic source code generation. She holds several patents and has published over two dozen papers in various areas, such as magnetic recording, error-correction coding,*

and control of robotic vehicles (for the Mars Pathfinder Rover).

Jose Salcedo *is a Chief Software Engineer working in the Network Technology Development group at JPL. He obtained his BA degree in physics from Occidental College and MS degree in Computer Engineer from USC.*

Tim Menzies *is a cognitive scientist exploring how quirks in human cognition effect the process of software and knowledge engineering. He holds a Ph.D. in artificial intelligence (1995), a masters of cognitive science (1988) and a computer science undergrad degree (1985), all from the University of New South Wales, Sydney, Australia. Currently, he leads the software assurance research at NASA's Independent Verification and Validation (IV&V) Facility. Prior to working with NASA, Dr. Menzies has worked with commercial organizations on object-oriented systems and expert systems. For further details, see http://tim.menzies.com*